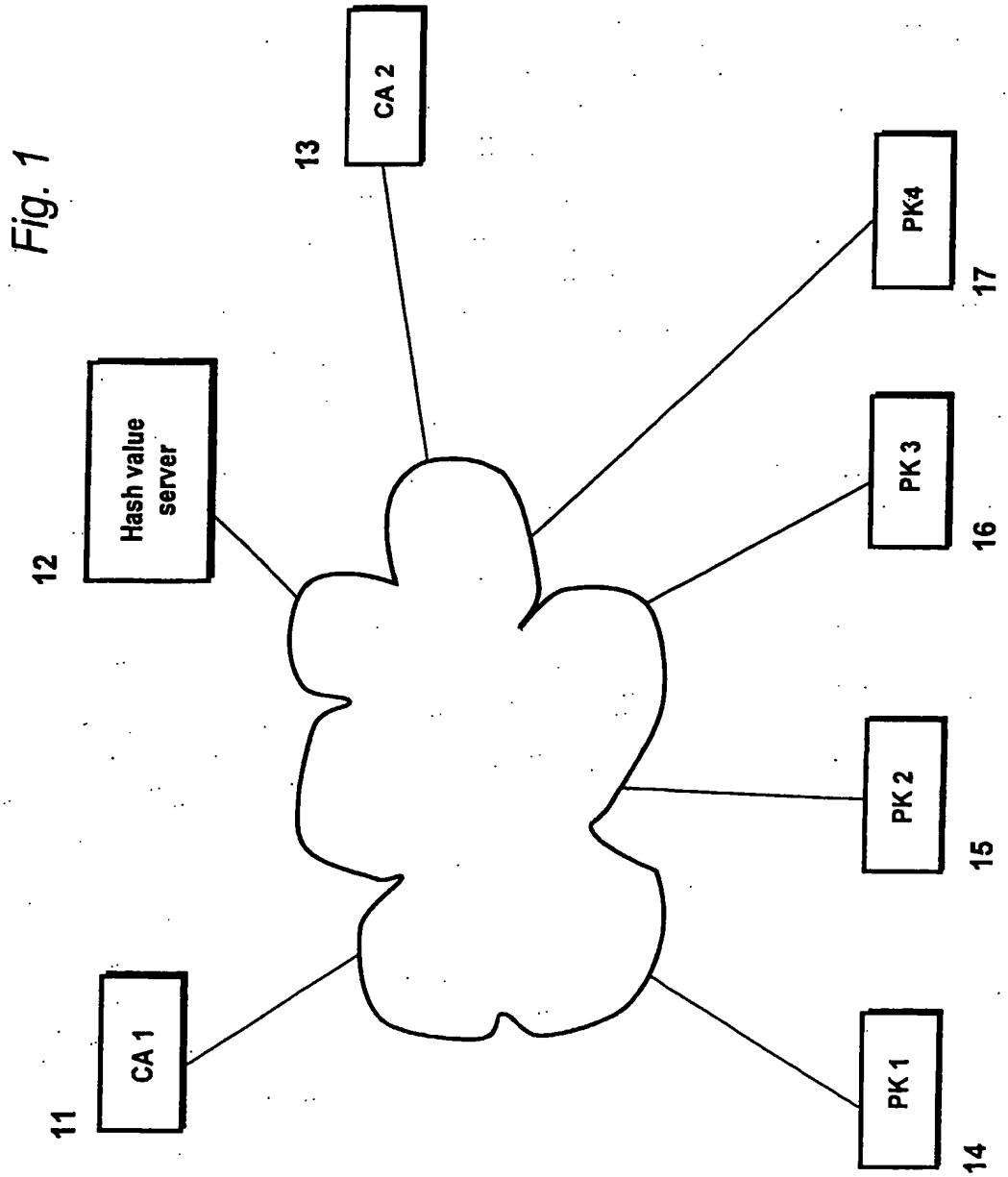
Fig. 1



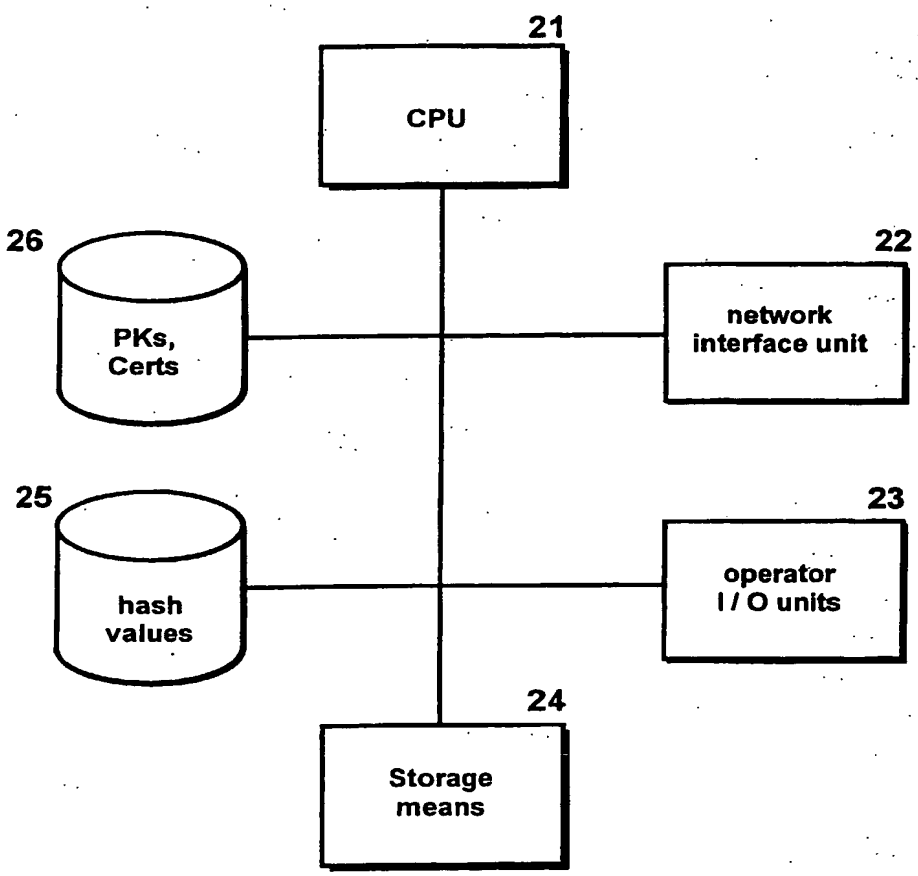

21
CPU
26
PKs,
Certs
22
network
interface unit
25
hash
values
23
operator
I / O units
24
Storage
means

Fig. 2

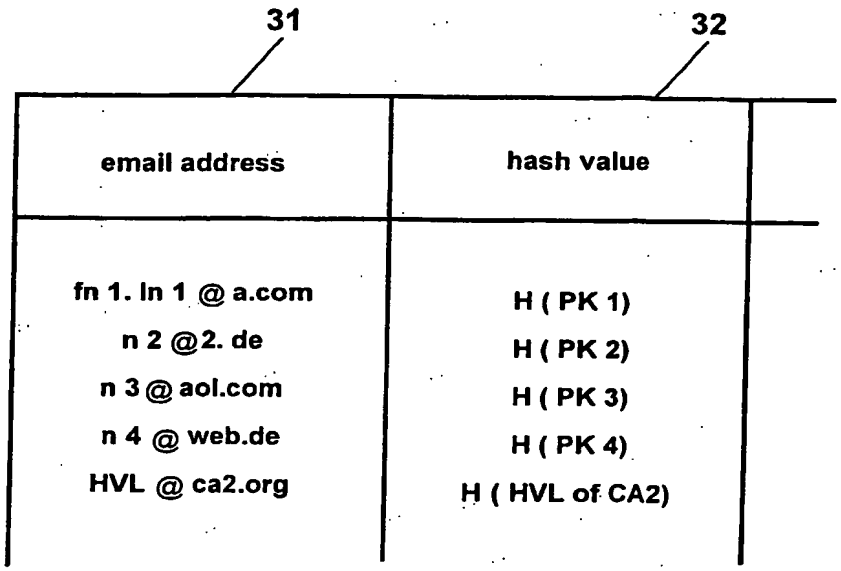

31
32
email address
hash value
fn 1. ln 1 @ a.com
n 2 @2. de
n 3 @ aol.com
n 4 @ web.de
HVL @ ca2.org
H (PK 1)
H (PK 2)
H (PK 3)
H (PK 4)
H (HVL of CA2)

Fig. 3

41 user ID	42 public key	43 certificate
1	PK 1	CA 1_cert (PK 1)
2	PK 2	CA 2_cert (PK 2)
3	PK 3	CA 1_cert (PK 3)
4	PK 4	CA 1_cert (PK 4)
CA 2	PK- CA 2	CA 1_cert (PK_CA 2)

Fig. 4

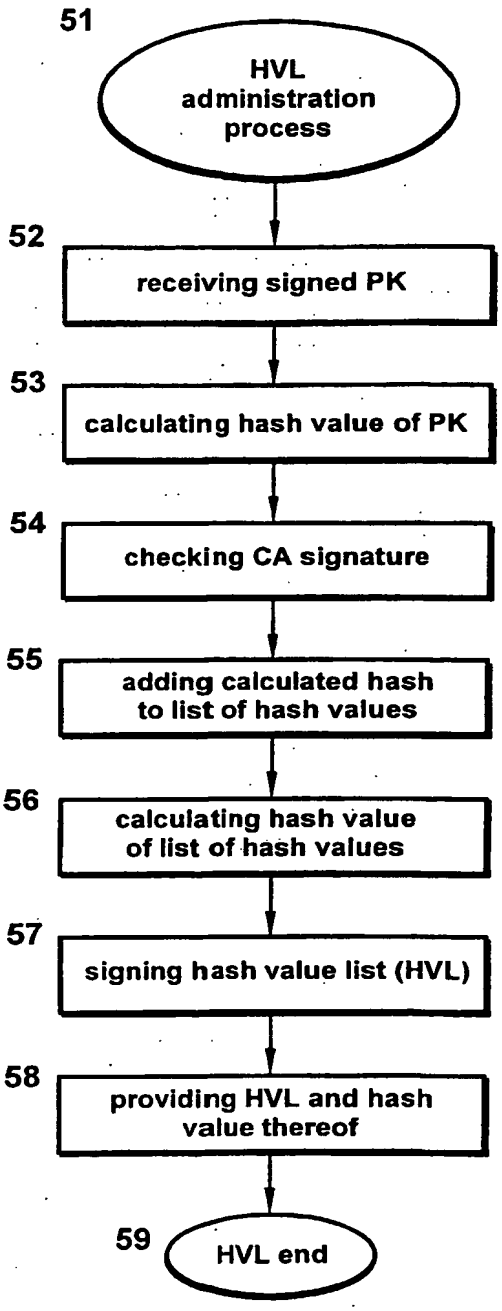

51
HVL administration process
52
receiving signed PK
53
calculating hash value of PK
54
checking CA signature
55
adding calculated hash to list of hash values
56
calculating hash value of list of hash values
57
signing hash value list (HVL)
58
providing HVL and hash value thereof
59
HVL end

Fig. 5

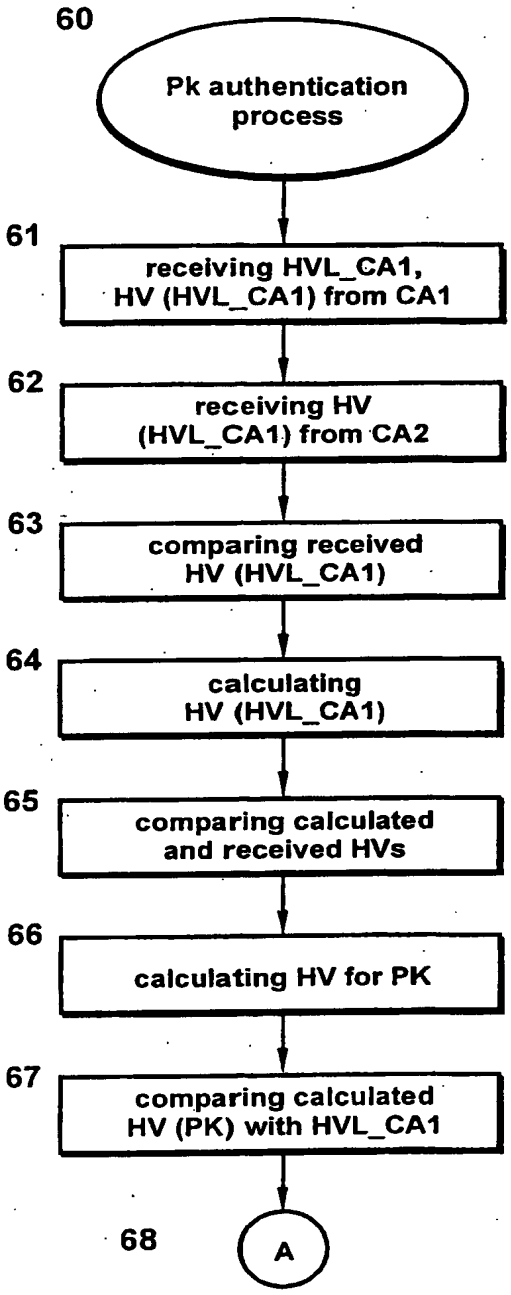

60
Pk authentication process
61
receiving HVL_CA1, HV (HVL_CA1) from CA1
62
receiving HV (HVL_CA1) from CA2
63
comparing received HV (HVL_CA1)
64
calculating HV (HVL_CA1)
65
comparing calculated and received HVs
66
calculating HV for PK
67
comparing calculated HV (PK) with HVL_CA1
68
A

Fig. 6

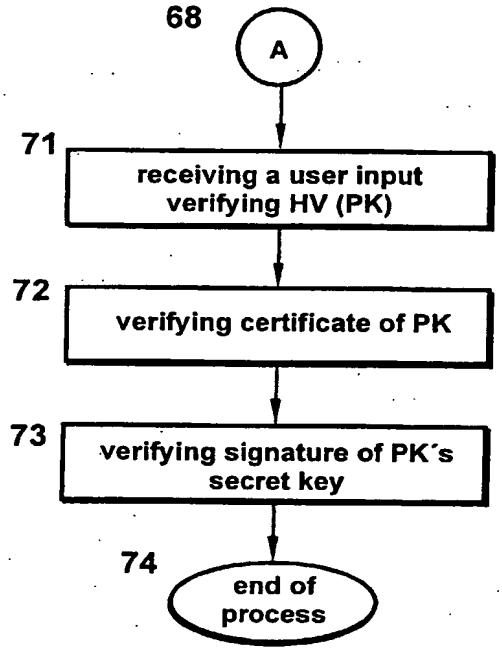

68
A
71
receiving a user input verifying HV (PK)
72
verifying certificate of PK
73
verifying signature of PK´s secret key
74
end of process

Fig. 7

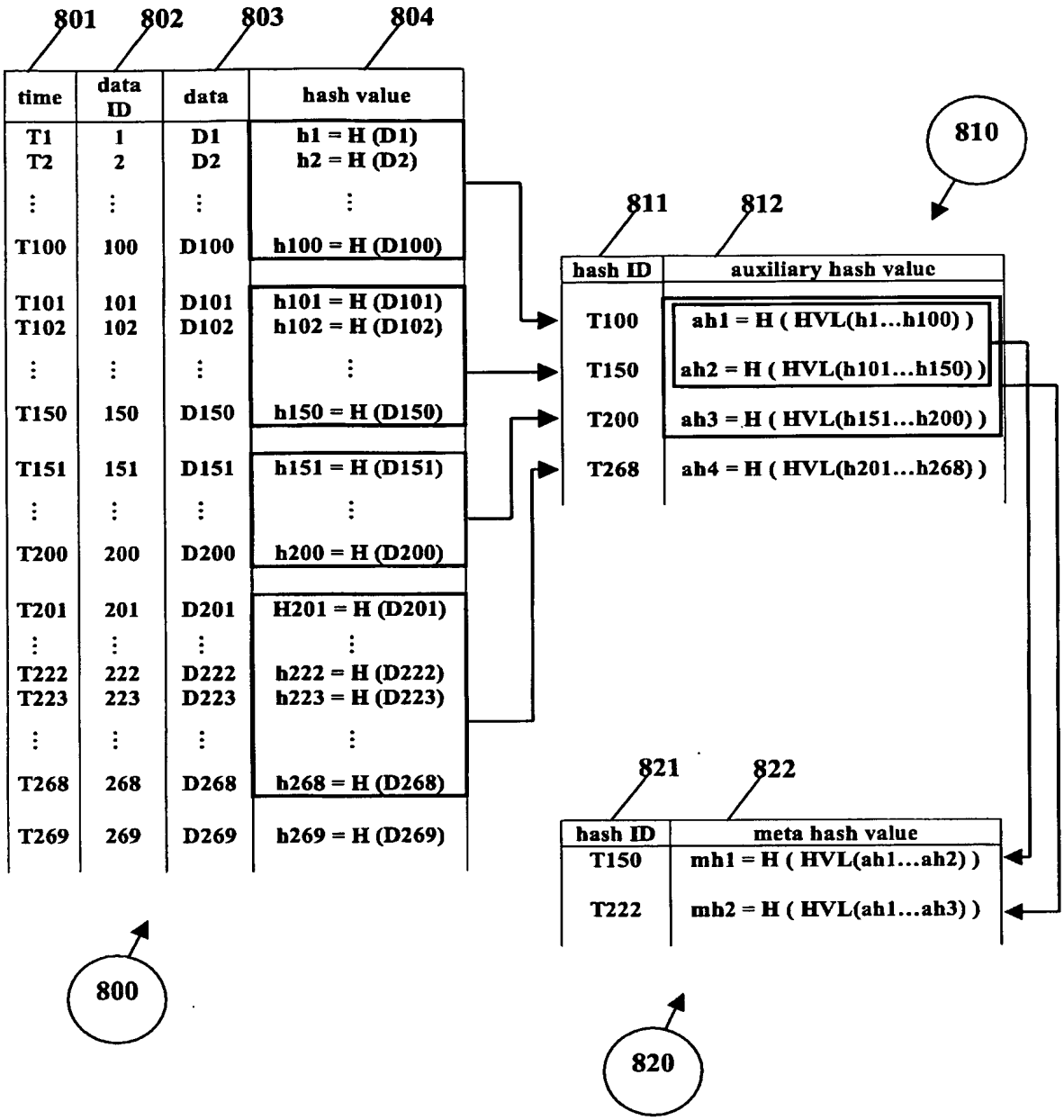

801
802
803
804
time
data ID
data
hash value
T1 1 D1 h1 = H (D1)
T2 2 D2 h2 = H (D2)
T100 100 D100 h100 = H (D100)
T101 101 D101 h101 = H (D101)
T102 102 D102 h102 = H (D102)
T150 150 D150 h150 = H (D150)
T151 151 D151 h151 = H (D151)
T200 200 D200 h200 = H (D200)
T201 201 D201 H201 = H (D201)
T222 222 D222 h222 = H (D222)
T223 223 D223 h223 = H (D223)
T268 268 D268 h268 = H (D268)
T269 269 D269 h269 = H (D269)
800
810
811
812
hash ID
auxiliary hash value
T100 ah1 = H (HVL(h1...h100))
T150 ah2 = H (HVL(h101...h150))
T200 ah3 = H (HVL(h151...h200))
T268 ah4 = H (HVL(h201...h268))
821
822
hash ID
meta hash value
T150 mh1 = H (HVL(ah1...ah2))
T222 mh2 = H (HVL(ah1...ah3))
820

Fig. 8

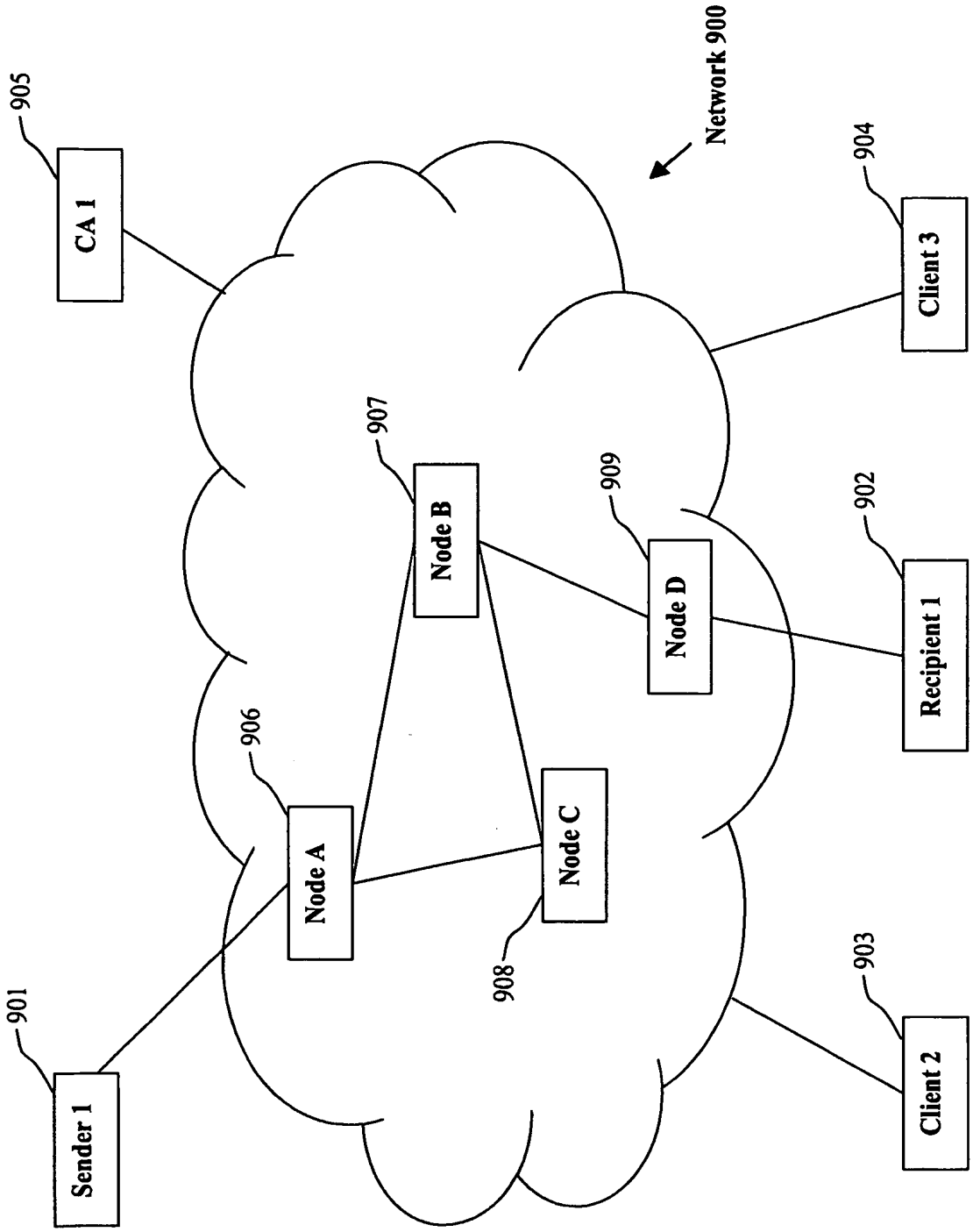

Network 900
905
CA 1
904
Client 3
907
Node B
909
Node D
902
Recipient 1
906
Node A
Node C
908
901
Sender 1
903
Client 2

Fig. 9

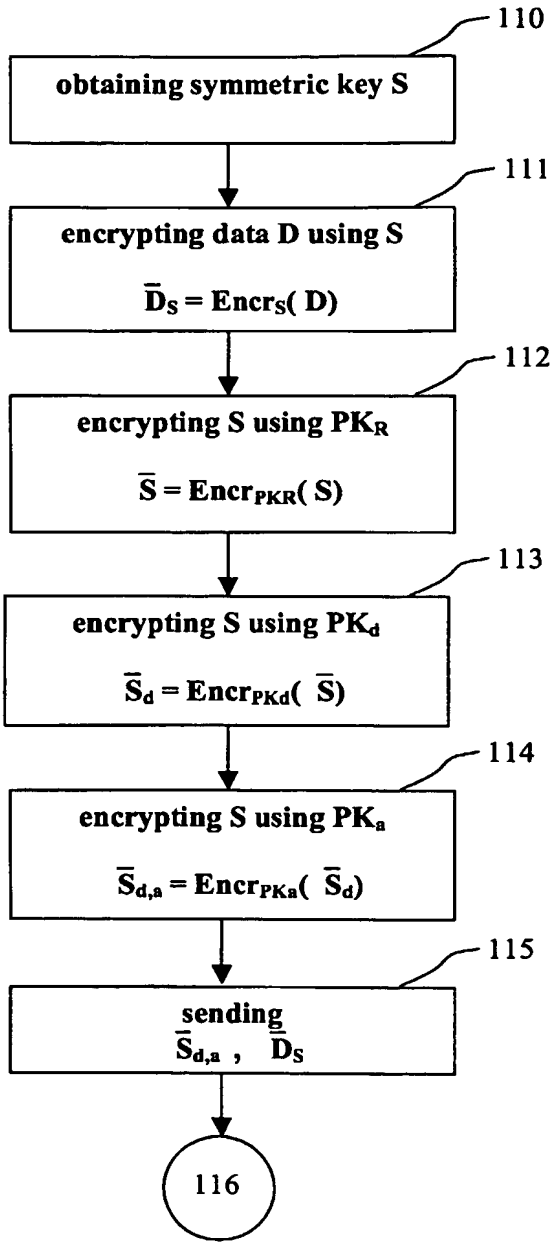

110
obtaining symmetric key S
111
encrypting data D using S
$\bar{D}_S = Encr_S(D)$
112
encrypting S using PK_R
$\bar{S} = Encr_{PKR}(S)$
113
encrypting S using PK_d
$\bar{S}_d = Encr_{PKd}(\bar{S})$
114
encrypting S using PK_a
$\bar{S}_{d,a} = Encr_{PKa}(\bar{S}_d)$
115
sending
$\bar{S}_{d,a}$, $\bar{D}_S$
116

Fig. 10

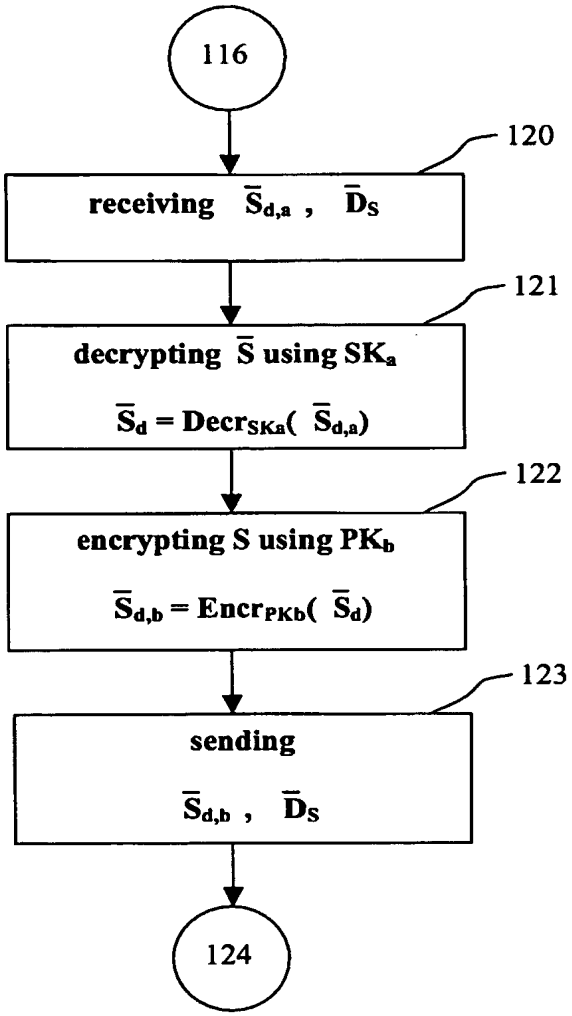

116
120
receiving $\bar{S}_{d,a}$, $\bar{D}_S$
121
decrypting $\bar{S}$ using SK_a
$\bar{S}_d = Decr_{SKa}(\bar{S}_{d,a})$
122
encrypting S using PK_b
$\bar{S}_{d,b} = Encr_{PKb}(\bar{S}_d)$
123
sending
$\bar{S}_{d,b}$, $\bar{D}_S$
124

Fig. 11

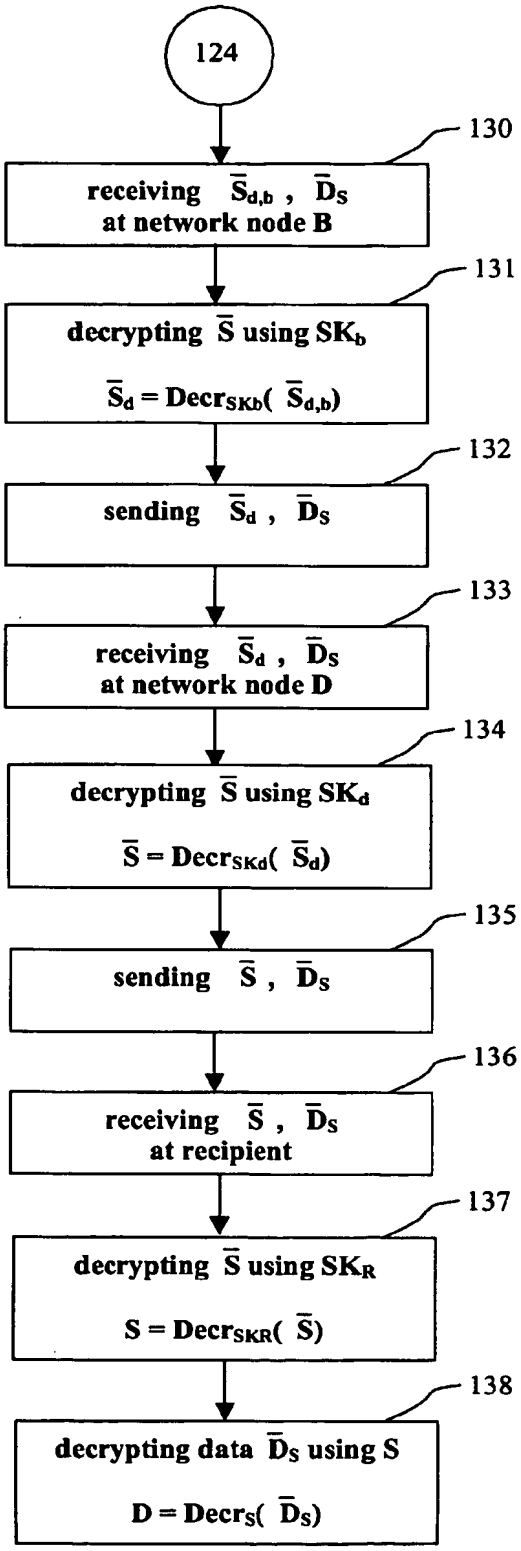

124
130
receiving $\bar{S}_{d,b}$, $\bar{D}_S$ at network node B
131
decrypting $\bar{S}$ using SK_b
$\bar{S}_d = Decr_{SKb}(\bar{S}_{d,b})$
132
sending $\bar{S}_d$, $\bar{D}_S$
133
receiving $\bar{S}_d$, $\bar{D}_S$ at network node D
134
decrypting $\bar{S}$ using SK_d
$\bar{S} = Decr_{SKd}(\bar{S}_d)$
135
sending $\bar{S}$, $\bar{D}_S$
136
receiving $\bar{S}$, $\bar{D}_S$ at recipient
137
decrypting $\bar{S}$ using SK_R
$S = Decr_{SKR}(\bar{S})$
138
decrypting data $\bar{D}_S$ using S
$D = Decr_S(\bar{D}_S)$

Fig. 12

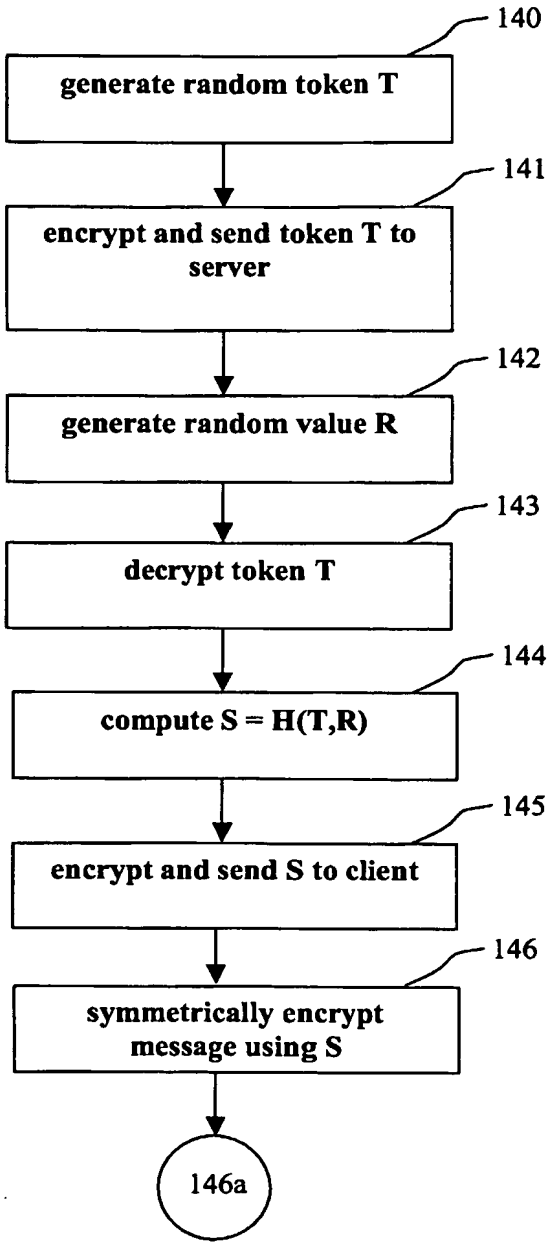

140
generate random token T
141
encrypt and send token T to server
142
generate random value R
143
decrypt token T
144
compute S = H(T,R)
145
encrypt and send S to client
146
symmetrically encrypt message using S
146a

Fig. 13a

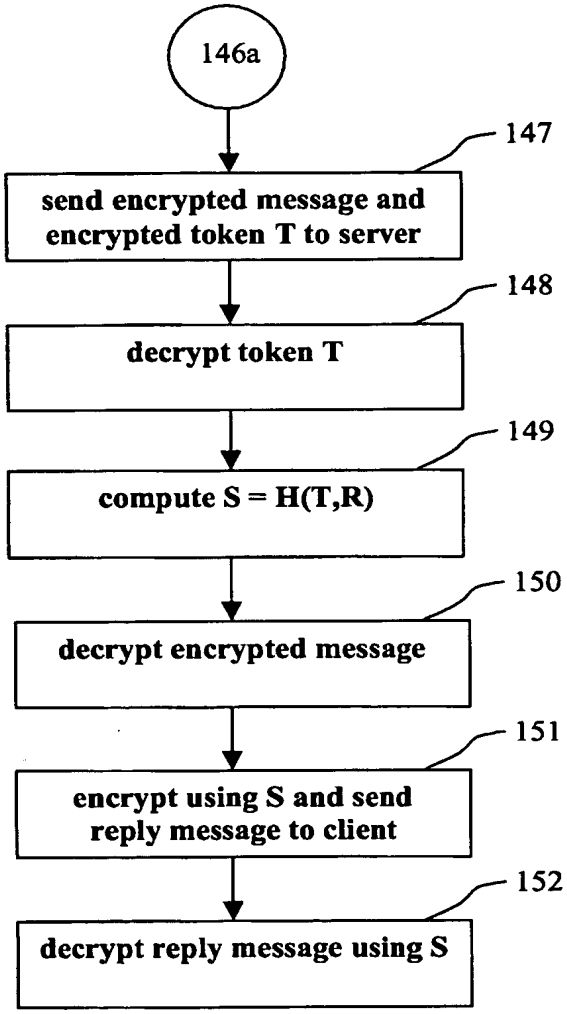

146a
147
send encrypted message and encrypted token T to server
148
decrypt token T
149
compute S = H(T,R)
150
decrypt encrypted message
151
encrypt using S and send reply message to client
152
decrypt reply message using S

Fig. 13b